7600-23

```
     (FILE 'USPAT' ENTERED AT 08:54:16 ON 24 AUG 1999)
L1          8375 S (SUBSEQUENT OR SUB OR PREVIOUS OR OLD OR SAME OR DIFFERE
NT)
L2            24 S ((VERSION OR REVISION) AND PROTOCOL#) /AB
L3         20383 S (IMPLEMENT? OR APPEND?) /AB
L4             2 S L2 AND L3
L5            34 S ALIAS(3A)(TAG# OR TAGGING)
L6          3094 S (ADDITION? OR ADDITIVE)(3A)(TAG# OR FLAG#)
L7          1848 S (ADDITION? OR ADDITIVE OR ADD##)(3A)(TAG#)
L8            25 S L5 AND L7
L9            24 S L8 AND PROTOCOL#
L10           24 S (IMPLEMENT? OR APPEND?) AND L9
L11       516767 S SUB(W)VERSION OR SUBSEQUENT
L12           23 S L10 AND L11
L13            0 S L2 AND L12
L14         7483 S (PROTOCOL# OR VERSION#) /AB
L15            0 S L12 AND L14
L16         4387 S (PLURALITY OR MORE OR DIFFERENT OR MULTIPLE)(3A)VERSION
OR
L17         1412 S (SUBSEQUEN? OR ADDITON? OR ADDITIVE OR PREVIOUS?)(3A)(VE
RSI
L18           51 S L16(P)L17
L19           14 S L14 AND L18
L20            1 S L19 AND TAG#

=> d


1.  5,893,908, Apr. 13, 1999, Document management system; John Cullen, et
al., 707/5; 345/348; 707/2, 500 [IMAGE AVAILABLE]

=> d l19 1-14


1.  5,909,581, Jun. 1, 1999, Automatic software updating method;
Seong-Kab Park, 395/712 [IMAGE AVAILABLE]

2.  5,893,908, Apr. 13, 1999, Document management system; John Cullen, et
al., 707/5; 345/348; 707/2, 500 [IMAGE AVAILABLE]

3.  5,878,432, Mar. 2, 1999, Object oriented framework mechanism for a
source code repository; David Joseph Misheski, et al., 707/103; 395/712;
707/104, 203 [IMAGE AVAILABLE]

4.  5,805,861, Sep. 8, 1998, Method of stabilizing component and net
names of integrated circuits in electronic design automation systems;
Douglas J. Gilbert, et al., 395/500.36, 500.19 [IMAGE AVAILABLE]

5.  5,649,200, Jul. 15, 1997, Dynamic rule-based version control system;
David B. Leblang, et al., 395/703; 364/222.81, 222.82, DIG.1; 707/203
[IMAGE AVAILABLE]

6.  5,600,834, Feb. 4, 1997, Method and apparatus for reconciling
different versions of a file; John H. Howard, 707/201; 364/281.5, 282.1,
282.4, DIG.1; 395/712 [IMAGE AVAILABLE]

7.  5,574,898, Nov. 12, 1996, Dynamic software version auditor which
monitors a process to provide a list of objects that are accessed; David
B. Leblang, et al., 707/1; 364/221.7, 280.6, DIG.1; 712/220 [IMAGE
```

AVAILABLE]

8. 5,493,668, Feb. 20, 1996, Multiple processor system having software for selecting shared cache entries of an associated castout class for transfer to a DASD with one I/O operation; David A. Elko, et al., 711/130; 364/228.1, 243.41, 243.7, 246.3, DIG.1; 711/113 [IMAGE AVAILABLE]

9. 5,481,713, Jan. 2, 1996, Method and apparatus for patching code residing on a read only memory device; Russ Wetmore, et al., 395/705, 712 [IMAGE AVAILABLE]

10. 5,388,493, Feb. 14, 1995, Extra low profile housing for vertical dual keyboard MIDI wireless controller for accordionists; Giorgio F. Curletto, 84/376A [IMAGE AVAILABLE]

11. 5,317,729, May 31, 1994, Method for the storage of multi-versioned data with retrieval based on searched query; Sujan K. Mukherjee, et al., 707/3; 364/282.1, 283.1, DIG.1; 707/203 [IMAGE AVAILABLE]

12. 5,287,473, Feb. 15, 1994, Non-blocking serialization for removing data from a shared cache; Chandrasekaran Mohan, et al., 711/133, 130 [IMAGE AVAILABLE]

13. 5,280,612, Jan. 18, 1994, Multiple version database concurrency control system; Raymond A. Lorie, et al., 707/8; 364/222.81, 222.82, 246.6, 246.8, 259, 259.2, 261, 262.4, 262.9, 264, 281.1, 281.3, 281.4, 281.5, 282.1, 282.4, DIG.1; 707/9, 203, 206; 710/242 [IMAGE AVAILABLE]

14. 4,860,352, Aug. 22, 1989, Satellite communication system and method with message authentication suitable for use in financial institutions; Joel E. Laurance, et al., 380/23; 340/825.34; 342/357.01, 450, 451; 455/12.1; 705/44 [IMAGE AVAILABLE]

```
=> s select?(s)(version# or revision#)(p)protocol#

       1407668 SELECT?
        120245 VERSION#
          8082 REVISION#
         63337 PROTOCOL#
L1         234 SELECT?(S)(VERSION# OR REVISION#)(P)PROTOCOL#

=> s subsequen?(a)version

        810372 SUBSEQUEN?
        101317 VERSION
L2          54 SUBSEQUEN?(A)VERSION

=> s (previous or late?)(a)version

        318764 PREVIOUS
        862553 LATE?
        101317 VERSION
L3        1096 (PREVIOUS OR LATE?)(A)VERSION

=> s l1 and l2 and l3

L4           1 L1 AND L2 AND L3

=> d


L4   ANSWER 1 OF 1  USPATFULL
AN     1999:168074  USPATFULL
TI     Systems and methods for automatic application version upgrading and
       maintenance
IN     Heath, Clifford, Mount Waverly, Australia
       Port, Graeme, Surrey Hills, Australia
       Klos, Steven, Nashua, NH, United States
       Greenhill, Graeme, Londonderry, NH, United States
PA     Open Software Associates, Ltd., Kingwood, Australia (non-U.S.
       corporation)
PI     US 6006034  19991221
AI     US 1996-707622  19960905 (8)
DT     Utility
LN.CNT 1142
INCL   INCLM: 395/712.000
NCL    NCLM:  395/712.000
IC     [6]
       ICM: G06F009-06
EXF    395/712; 395/200.51; 395/200.5; 395/200.47; 395/200.49; 395/500;
       395/181; 707/203; 707/507; 707/200; 380/4; 370/316


=> d kwic


L4   ANSWER 1 OF 1  USPATFULL
SUMM   . . . kept current by replacing one or more of such components, or
by
       adding or deleting components. The components having the latest
     version numbers can be maintained on the central server and
       distributed from the server to each individual client as needed
through.
```

. .

SUMM With the method of the present invention, controlling a **version** upgrade at the client through an open network environment, such as the Internet, can be easily implemented. In the Internet. . . integrated into the browser as a helper application, a plug-in module, or as a browser control. When the link is **selected** from the client browser, the launcher is executed to update the corresponding application components on the client. The downloading chores of the update can be accomplished through standard file transfer methods such as the file transfer **protocol** or the hypertext transfer **protocol**. The catalog file can also be specified to include a procedure to install on the client desktop an icon or. . .

DETD . . . preferred embodiment, the server responds by downloading a catalog of a list of the application components, each identified with the **latest version** number. Here, the server includes either a single computer or multiple computers or other servers networked together. The catalog file. . .

DETD . . . which at least includes the updated list of components and version numbers on the client, for a comparison in a **subsequent version** check. The components may either be stored in cache 22a or in program directories, such as 22b to 22d, specified. . .

DETD . . . required, the catalog file includes at 324 a version identification, code or data size, and the network address(es) where the **latest version** of the component is stored. The components themselves may also be included within the catalog file in certain updating situations.. . .

DETD . . . list of components and version numbers on the client, is stored at 317 in cache on the client until the **subsequent version** update.

CLM What is claimed is:

. . . to a subsequent call to the server from the client, causing the server to download a second catalog including the **latest version** identifications of the components and of any new additional components on the server; comparing in the client the **latest version** identifications of the components in the second catalog with the version identifications of the components maintained on the client; updating. . .

=> s premature(3a)tag#

```
         46781 PREMATURE
         26308 TAG#
L1          10 PREMATURE(3A)TAG#
```

=> d 1-10 kwic

L1 ANSWER 1 OF 10 USPATFULL
DETD . . . the fifth family from Tunisia (ST) was identified as a G to T transversion at nucleotide position G39 creating a **premature** stop codon (GAG>**TAG**) at codon 47, and was designated E47X. In each family, normal hearing parents were found to be heterozygous for the. . .

L1 ANSWER 2 OF 10 USPATFULL
DETD . . . mutation was a single base deletion of a cytosine at nucleotide 1737, resulting in a frame shift that created a **premature** stop codon **TAG**, resulting in the removal of 80 amino acids (a deletion of all amino acids after amino acid 463). A sixth. . .

L1 ANSWER 3 OF 10 USPATFULL
SUMM . . . electrostatic discharge protection during tag manufacture is that described by Benge et al. in U.S. Pat. No. 4,910,499 for avoiding **premature** deactivation of the **tags** during printing, in which a film of electrostatic-charge-draining material is applied to the outside of a release liner overlaying a. . .

L1 ANSWER 4 OF 10 USPATFULL
DETD . . . mutation was a single base deletion of a cytosine at nucleotide 1737, resulting in a frame shift that created a **premature** stop codon **TAG**, resulting in the removal of 80 amino acids (a deletion of all amino acids after amino acid 463). Another mutation. . .

L1 ANSWER 5 OF 10 USPATFULL
SUMM While the cuts 520 also have the effect of preventing **premature** deactivation in the **tag** manufacturing equipment or subsequently in printing equipment due to electrostatic charge, applicant looks to the same, as in the case. . .

L1 ANSWER 6 OF 10 USPATFULL
DETD It should be noted that the cuts 520 also have the effect of preventing **premature** deactivation in the **tag** manufacturing equipment or subsequently in printing equipment due to electrostatic discharge.

L1 ANSWER 7 OF 10 USPATFULL
AB . . . pair are connected by welding to provide a reliable circuit. A film of electrostatic-charge-draining material on a web of deactivatable **tags** prevents their **premature** deactivation.

L1 ANSWER 8 OF 10 USPATFULL
AB . . . pair are connected by welding to provide a reliable circuit. A film of electrostatic-charge-draining material on a web of deactivatable

tags prevents their **premature** deactivation.

L1 ANSWER 9 OF 10 USPATFULL
DETD . . . also been found possible to combine the antibody binding protein and the tag hormone in the same gel particles, without **premature** binding of the **tag** hormone to antibody binding sites, thereby to eliminate the need for separate preparation of the gel particles and to form. . .
DETD . . . a non-aqueous medium. Thus incorporation of the tag hormone is carried out when the antibody-gel particles are dry--which should preclude **premature** binding of the **tag** hormone to antibody binding sites.

L1 ANSWER 10 OF 10 USPATFULL
DETD . . . to combine the antibody binding protein and a tag hormone or other tagged material in the same gel particles, without **premature** binding of the **tag** hormone to antibody binding sites. For this purpose, the binding protein is entrapped in the dry gel particles in the. . .

=> d 1-10

L1 ANSWER 1 OF 10 USPATFULL
AN 1999:159763 USPATFULL
TI Mutated polynucleotide corresponding to a mutation responsible for prelingual non-syndromic deafness in the connexin 26 gene and method of detecting this hereditary defect
IN Petit, Christine, Le Plessis-Robinson, France
Denoyelle-Gryson, Francoise, Arcueil, France
Weil, Dominique, Paris, France
Marlin-Duvernois, Sandrine, Colombes, France
Guesdon, Jean-Luc, Sevres, France
PA Institut Pasteur, Paris, France (non-U.S. corporation)
PI US 5998147 19991207
AI US 1998-134566 19980814 (9)
PRAI US 1997-55863 19970815 (60)
DT Utility
LN.CNT 1119
INCL INCLM: 435/006.000
INCLS: 435/091.200; 536/024.300
IC [6]
ICM: C12Q001-68
ICS: C12P019-34; C07H021-04
EXF 435/6; 435/91.2; 536/24.3
CAS INDEXING IS AVAILABLE FOR THIS PATENT.

L1 ANSWER 2 OF 10 USPATFULL
AN 1999:121133 USPATFULL
TI Diagnosis and treatment of glaucoma
IN Sarfarazi, Mansoor, New Britain, CT, United States
PA The University of Connecticut, Farmington, CT, United States (U.S. corporation)
PI US 5962230 19991005
AI US 1997-926492 19970910 (8)
RLI Continuation-in-part of Ser. No. US 1997-800036, filed on 13 Feb 1997, now patented, Pat. No. US 5830661
DT Utility
LN.CNT 1267
INCL INCLM: 435/006.000
INCLS: 435/091.100; 435/091.200; 536/023.100; 536/024.310; 536/024.330; 935/077.000; 935/078.000
NCL NCLM: 435/006.000

NCLS: 435/091.100; 435/091.200; 536/023.100; 536/024.310; 536/024.330
IC [6]
ICM: C07H021-04
ICS: C07H021-02; C12Q001-68; C12P019-34
EXF 435/6; 435/91.2; 435/91.1; 536/23.1; 536/24.31; 536/24.33; 935/77; 935/78
CAS INDEXING IS AVAILABLE FOR THIS PATENT.

L1 ANSWER 3 OF 10 USPATFULL
AN 1998:148053 USPATFULL
TI Electronic security tag useful in electronic article indentification and surveillance system
IN Appalucci, Lawrence, Villanova, PA, United States
Bowers, John H., Clarksburg, NJ, United States
Mazoki, Gary T., Sewell, NJ, United States
McKeown, Thomas J., Pennsauken, NJ, United States
Piccoli, Anthony F., Audubon, NJ, United States
Rankin, Mark J., Media, PA, United States
Tocker, Stanley, Wilmington, DE, United States
PA Checkpoint Systems, Inc., Thorofare, NJ, United States (U.S. corporation)
PI US 5841350 19981124
AI US 1997-884409 19970627 (8)
DT Utility
LN.CNT 1035
INCL INCLM: 340/572.000
INCLS: 524/910.000
NCL NCLM: 340/572.300
NCLS: 340/572.500; 524/910.000
IC [6]
ICM: G08B013-14
EXF 340/572; 361/1; 361/56; 361/91; 361/117; 361/118; 257/173; 257/355; 427/96; 427/123; 524/910

L1 ANSWER 4 OF 10 USPATFULL
AN 1998:134817 USPATFULL
TI Diagnosis and treatment of glaucoma
IN Sarfarazi, Mansoor, New Britain, CT, United States
PA The University of Connecticut, Storrs, CT, United States (U.S. corporation)
PI US 5830661 19981103
AI US 1997-800036 19970213 (8)
DT Utility
LN.CNT 1190
INCL INCLM: 435/006.000
INCLS: 435/091.200; 536/024.310; 935/077.000; 935/078.000
NCL NCLM: 435/006.000
NCLS: 435/091.200; 536/024.310
IC [6]
ICM: C12Q001-68
ICS: C12P019-34; C07H021-04; C07H021-02
EXF 435/6; 435/91.2; 435/875; 435/91.1; 536/24.31; 935/77; 935/78
CAS INDEXING IS AVAILABLE FOR THIS PATENT.

L1 ANSWER 5 OF 10 USPATFULL
AN 96:16634 USPATFULL
TI Methods for the making of electronic article surveillance tags and improved electronic article surveillance tags produced thereby
IN Benge, S. Eugene, Middletown, OH, United States
PA Sensormatic Electronics Corporation, Deerfield Beach, FL, United States (U.S. corporation)
PI US 5494550 19960227
AI US 1993-117785 19930907 (8)
DT Utility
LN.CNT 823

INCL INCLM: 156/268.000
INCLS: 156/051.000; 156/052.000; 156/247.000; 156/248.000; 156/256.000; 156/265.000; 156/270.000; 156/324.000; 340/572.000
NCL NCLM: 156/268.000
NCLS: 156/051.000; 156/052.000; 156/247.000; 156/248.000; 156/256.000; 156/265.000; 156/270.000; 156/324.000; 340/572.300
IC [6]
ICM: B32B031-04
ICS: B32B031-18; G08B013-14
EXF 156/51; 156/52; 156/268; 156/324; 156/256; 156/270; 156/247; 156/248; 156/252; 156/265; 340/572

L1 ANSWER 6 OF 10 USPATFULL
AN 91:28922 USPATFULL
TI Electronic article surveillance tag and method of deactivating tags
IN Benge, S. Eugene, Middletown, OH, United States
Froning, Robert L., Kettering, OH, United States
PA Monarch Marking Systems, Inc., Dayton, OH, United States (U.S. corporation)
PI US 5006856 19910409
AI US 1989-397804 19890823 (7)
DT Utility
LN.CNT 868
INCL INCLM: 340/572.000
NCL NCLM: 340/572.300
IC [5]
ICM: G08B013-14
EXF 340/572; 343/894; 343/895

L1 ANSWER 7 OF 10 USPATFULL
AN 90:21852 USPATFULL
TI Tag and method of making same
IN Benge, S. Eugene, Middletown, OH, United States
Froning, Robert L., Kettering, OH, United States
PA Monarch Marking Systems, Inc., Dayton, OH, United States (U.S. corporation)
PI US 4910499 19900320
AI US 1989-308699 19890210 (7)
RLI Continuation-in-part of Ser. No. US 1987-114792, filed on 28 Oct 1987, now patented, Pat. No. US 4818312 which is a division of Ser. No. US 1987-124712, filed on 24 Nov 1987, now patented, Pat. No. US 4846922, issued on 11 Jul 1989 which is a continuation-in-part of Ser. No. US 1987-81096, filed on 3 Aug 1987 which is a continuation-in-part of Ser. No. US 1987-41556, filed on 22 Apr 1987 which is a continuation-in-part of Ser. No. US 1986-912466, filed on 29 Sep 1986
DT Utility
LN.CNT 715
INCL INCLM: 340/572.000
INCLS: 156/324.000; 156/051.000; 156/052.000; 361/402.000; 427/096.000; 427/121.000; 427/123.000
NCL NCLM: 340/572.300
NCLS: 156/051.000; 156/052.000; 156/324.000; 340/572.500; 361/765.000; 427/096.000; 427/121.000; 427/123.000
IC [4]
ICM: G08B013-24
EXF 340/572; 156/52; 156/155; 156/272.2; 156/273.9; 156/324; 156/51; 361/402; 427/96; 427/121; 427/123; 333/185

L1 ANSWER 8 OF 10 USPATFULL
AN 89:56072 USPATFULL
TI Method of making deactivatable tags
IN Benge, S. Eugene, Middletown, OH, United States
Froning, Robert L., Kettering, OH, United States
PA Monarch Marking Systems, Inc., Dayton, OH, United States (U.S. corporation)
PI US 4846922 19890711

AI US 1987-124712 19871124 (7)
RLI Continuation-in-part of Ser. No. US 1987-114792, filed on 28 Oct 1987 which is a continuation-in-part of Ser. No. US 1987-81096, filed on 3 Aug 1987, now abandoned which is a continuation-in-part of Ser. No. US 1987-41556, filed on 22 Apr 1987, now patented, Pat. No. US 4778552 which is a continuation-in-part of Ser. No. US 1986-912466, filed on 29 Sep 1986, now patented, Pat. No. US 4717438
DT Utility
LN.CNT 711
INCL INCLM: 156/324.000
INCLS: 156/051.000; 156/052.000; 340/572.000; 361/402.000; 427/096.000; 427/121.000; 427/123.000
NCL NCLM: 156/324.000
NCLS: 156/051.000; 156/052.000; 340/572.300; 340/572.500; 361/765.000; 427/096.000; 427/121.000; 427/123.000
IC [4]
ICM: B32B031-08
ICS: B32B031-10; B32B031-12
EXF 156/52; 156/155; 156/272.2; 156/273.9; 156/324; 156/51; 333/185; 340/572; 361/402; 427/96; 427/121; 427/123

L1 ANSWER 9 OF 10 USPATFULL
AN 79:6941 USPATFULL
TI Method and device for immunoassay
IN Updike, Stuart J., Madison, WI, United States
PA Wisconsin Alumni Research Foundation, Madison, WI, United States (U.S. corporation)
PI US 4138474 19790206
AI US 1973-356092 19730501 (5)
DT Utility
LN.CNT 437
INCL INCLM: 424/001.000
INCLS: 023/230.000B; 023/230.600; 424/012.000; 210/031.000C; 210/504.000; 422/058.000; 422/101.000
NCL NCLM: 436/535.000
NCLS: 210/504.000; 210/635.000; 422/058.000; 422/101.000; 436/542.000; 436/810.000
IC [2]
ICM: G01N033-16
ICS: A61K043-00
EXF 023/230B; 023/259; 424/1; 424/12; 252/316; 210/31C; 210/504
CAS INDEXING IS AVAILABLE FOR THIS PATENT.

L1 ANSWER 10 OF 10 USPATFULL
AN 75:66889 USPATFULL
TI Preparation of dry, porous gel particles having high water regain for liquid sampling
IN Updike, Stuart J., Madison, WI, United States
PA Wisconsin Alumni Research Foundation, Madison, WI, United States (U.S. corporation)
PI US 3925017 19751209
AI US 1973-356093 19730501 (5)
DT Utility
LN.CNT 243
INCL INCLM: 023/230.000B
INCLS: 210/031.000C; 260/080.300N
NCL NCLM: 521/063.000
NCLS: 210/635.000; 436/180.000; 521/064.000; 526/306.000
IC [2]
ICM: B01D039-04
ICS: G01N031-06
EXF 023/230B; 023/259; 424/1; 424/12; 252/316; 210/31C; 210/504; 260/80.3N
CAS INDEXING IS AVAILABLE FOR THIS PATENT.

```
=> s premature(3a)(tag? or block?)

         46781 PREMATURE
         36557 TAG?
        827501 BLOCK?
L2         251 PREMATURE(3A)(TAG? OR BLOCK?)

=> s premature(3a)(block?)

         46781 PREMATURE
        827501 BLOCK?
L3         241 PREMATURE(3A)(BLOCK?)

=> s premature(a)(end)

         46781 PREMATURE
       1642533 END
L4          70 PREMATURE(A)(END)

=> s premature(a)(end)(a)block#

         46781 PREMATURE
       1642533 END
        719378 BLOCK#
L5           0 PREMATURE(A)(END)(A)BLOCK#

=> s l4(p)(data or information)

        549032 DATA
        436183 INFORMATION
L6          14 L4(P)(DATA OR INFORMATION)

=> d 1-14 kwic


L6   ANSWER 1 OF 14  USPATFULL
DETD   Valid data is considered to be present until either squelch
       level has not been generated for a time longer than 150 ns, indicating
       End of Packet. Once good data has been detected the squelch
       levels are reduced to minimize the effect of noise causing
     premature End of Packet detection.

L6   ANSWER 2 OF 14  USPATFULL
DETD   These turnovers are used by the transaction processor to sort out
       whether data has been properly and completely received from
       the field receiver. Because data from field receivers can be
       received by the data center in any order, it is necessary to
       track the data in some manner to determine if all current
     data has been received from all the field receivers. A normal
       stream of log file data from the field receiver will contain
       turnover packets with turnover numbers in ascending order. If any
       missing numbers or restarts or premature end-of-file
       codes indicate that either log files are missing or data has
       been lost on the field receiver. The transaction processor keeps track
       of any restarts or premature end-of-file condition
       found in log files. In addition, the minimum and maximum turnover
       numbers are stored for a particular range of.  .  .

L6   ANSWER 3 OF 14  USPATFULL
```

DETD After each of these conditions has been satisfied, twisted pair receiver
28 transmits the twisted pair input **data** signal RXTP and generates the valid **data** present signal VD, which indicates that valid **data** is present. When the valid **data** present signal VD is generated, twisted pair receiver 28 is reset and the squelch levels are reduced to minimum levels Vsq+/- to minimize the effect of noise which can produce a **premature end** of packet determination. Twisted pair receiver 28 determines a valid end of
packet condition when the twisted pair received input. . .

L6 ANSWER 4 OF 14 USPATFULL
SUMM Furthermore, the process in which the conventional CAD system calls on the operator to manually correct relevant CAD **data** until sufficient clearance values are attained is tedious, time-consuming and inefficient. Inadvertent operations by the operator may bring the preparation of the CAD **data** to a **premature end** without sufficient clearance values being attained. In such a case, the insufficient clearance values are recognized only after manufacture of. . .

L6 ANSWER 5 OF 14 USPATFULL
DETD After each of these conditions has been satisfied, twisted pair receiver
28 transmits the twisted pair input **data** signal RXTP and generates the valid **data** present signal VD, which indicates that valid **data** is present. When the valid **data** present signal VD is generated, twisted pair receiver 28 is reset and the squelch levels are reduced to minimum levels Vsq.+-. to minimize the
effect of noise which can produce a **premature end** of packet determination. Twisted pair receiver 28 determines a valid end of
packet condition when the twisted pair received input. . .

L6 ANSWER 6 OF 14 USPATFULL
DETD These turnovers are used by the transaction processor to sort out whether **data** has been properly and completely received from the field receiver. Because **data** from field receivers can be received by the **data** center in any order, it is necessary to track the **data** in some manner to determine if all current **data** has been received from all the field receivers. A normal stream of log file **data** from the field receiver will contain turnover packets with turnover numbers in ascending order. If any missing numbers or restarts or **premature end**-of-file codes indicate that either log files are missing or **data** has been lost on the field receiver. The transaction processor keeps track of any restarts or **premature end**-of-file condition found in log files. In addition, the minimum and maximum turnover numbers are stored for a particular range of. . .

L6 ANSWER 7 OF 14 USPATFULL
DETD The end-of-scan flag in bit 15 of the second transition **data** word (accumulator ACC2) is tested to determine whether the word represents a valid transition or an end-of-scan delimiter (Step 618). If
the second transition word represents an end-of-scan delimiter, a **premature end**-of-scan has occurred (i.e., a light-to-dark transition occurs without a corresponding dark-to-light transition). Such a **premature end**-of-scan is illustrated by the contents of locations 4 and 5 of RAM 142 (See FIG. 6A). Where a transition has. . .

L6 ANSWER 8 OF 14 USPATFULL
DETD . . . the volume measurement, i.e., in the interval between closure

of switch 90 and switch 92, switch 84 closes indicating a **premature end** to the test run. Switch 84 applies a high level to the **data** input of flip flop 96. This drives the Q output of flip flop 96 high thereby signalling the occurrence of. . .

L6 ANSWER 9 OF 14 USPATFULL

DETD The end-of-scan flag in bit 15 of the second transition **data** word (accumulator ACC2) is tested to determine whether the word represents a valid transition or an end-of-scan delimiter (Step 618). If the second transition word represents an end-of-scan delimiter, a **premature end**-of-scan has occurred (i.e., a light-to-dark transition occurs without a corresponding dark-to-light transition). Such a **premature end**-of-scan is illustrated by the contents of locations 4 and 5 of RAM 142 (See FIG. 6A). Where a transition has. . .

L6 ANSWER 10 OF 14 USPATFULL

DETD . . . and `0D`, FIGS. 6A, 6C, instruction loads the range register associated with the referenced channel. The unsigned, 16-bit, quantity loaded (**data** bus) is the number of bytes (range) to be transmitted during the **data** transfer that is being set up. The number is a positive binary quantity and is decremented by the UPC 201. . . transfer. The function code 0D is also a Go Function for printer adapters. A range of Zero results in a **premature end** -of-operation termination for any read or write command that may be subsequently issued. Any range register residue is applied to the. . .

L6 ANSWER 11 OF 14 USPATFULL

SUMM The invention provides the possibility of avoiding irreversible **data** losees of magnetic disk storage systems by monitoring, measuring and analyzing the triboelectric current, and, based on the analyzing results, initiating a **data** transfer shortly before the end of the lifetime of the **data** storage system. Thus the loss of valuable **data** can be avoided as well as avoiding a **premature end** to the service life of a **data** storage system.

L6 ANSWER 12 OF 14 USPATFULL

DETD . . . having memory windows asymmetrically distributed around the zero volt level. To avoid unduly shrinking the memory window and establishing a **premature end** of life condition it becomes necessary to use a bias voltage, herein VMREF, during the cell read operation to shift. . . thereby center the memory window. Consequently, both the memory window and bias voltage are important measurements in determining a reasonable **data** retention period.

L6 ANSWER 13 OF 14 USPATFULL

CLM What is claimed is:

4. The method of suspending a burst mode device initiated **data** transfer on a computer input/output channel for programmed I/O device of a programmed I/O device comprising the steps of: generating. . . request signal in said programmed I/O device; transmitting said instruction priority request signal to said burst mode device; generating a **premature end** of chain signal in said burst mode device in response to said instruction priority request; transmitting said end of chain. . . channel from burst mode to programmed I/O mode; terminating said instruction priority request signal thereby allowing the suspended burst mode **data** transfer to resume.

L6 ANSWER 14 OF 14 USPATFULL
DETD As a feature mentioned previously, the picking up of the handset at the station will terminate **data** transmission. The means for acomplishing this reaction is as follows: Normally the voltage on the telephone line as indicated by. . . on lead 396 restores latch gate 202 shutting off power to the system on lead 192 and thereby producing a **premature end** of transmission. The central **data** processor is designed to erase all **data** transmitted when less than the full message bit stream has been received. The shift registers reset and the stream counters 210 and 214 restore also. As mentioned previously, the chances of this occurring are slim, since **data** transmission will be completed in about one second, the **data** transmission occurring only once a week or once a month, at off-peak hours.

=> d 1-14

L6 ANSWER 1 OF 14 USPATFULL
AN 1998:55991 USPATFULL
TI Combination of input output circuitry and local area network systems
IN Davis, Timothy D., Arlington, TX, United States
Baker, Roman, Sunnyvale, CA, United States
Daugherty, Dan E., Burleson, TX, United States
Michael, Martin S., Los Gatos, CA, United States
Masood, Ahmed, Austin, TX, United States
Waterson, Kent Bruce, Everman, TX, United States
Fung, Hon C., Arlington, TX, United States
Koether, Mark Douglas, Grand Prairie, TX, United States
Johnson, J. Scott, Fort Worth, TX, United States
PA National Semiconductor Corp., Santa Clara, CA, United States (U.S. corporation)
PI US 5754764 19980519
AI US 1994-200097 19940222 (8)
DT Utility
LN.CNT 7858
INCL INCLM: 395/200.010
INCLS: 305/800.000; 305/412.000; 305/497.010; 305/876.000; 305/821.000; 305/555.000
NCL NCLM: 709/200.000
NCLS: 710/005.000; 710/056.000; 711/170.000; 711/202.000; 713/500.000
IC [6]
ICM: G06F013-00
ICS: H04L012-00
EXF 395/800; 395/200; 395/250; 395/275; 364/DIG.1; 375/62

L6 ANSWER 2 OF 14 USPATFULL
AN 97:59727 USPATFULL
TI System and method for monitoring video program material
IN Copriviza, Robert C., Tarzana, CA, United States
Dubin, Arnold M., Calabasas, CA, United States
Ackerman, Edward B., Encino, CA, United States
Wood, Jackson B., Tarzana, CA, United States
Eakins, Jeffrey S., Claremont, CA, United States
Harmon, David D., Torrance, CA, United States
PA Airtrax, United States (U.S. corporation)
PI US 5646675 19970708
AI US 1994-255222 19940606 (8)
RLI Division of Ser. No. US 1989-370399, filed on 22 Jun 1989, now patented, Pat. No. US 5319453, issued on 7 Jun 1994
DT Utility
LN.CNT 2896

INCL INCLM: 348/001.000
INCLS: 455/002.000
NCL NCLM: 348/001.000
NCLS: 455/002.000
IC [6]
ICM: H04N007-00
EXF 348/1; 348/2; 348/4; 348/3; 348/5; 348/473; 348/476-479; 455/2; 358/188;
358/85; 358/86; 358/84; 358/83; 358/142; 360/14.1; 360/14.2; 360/14.3; 360/33.1; H04N007-08; 7087; 700; 716; 7167; 7175; 718

L6 ANSWER 3 OF 14 USPATFULL
AN 96:37360 USPATFULL
TI Twisted pair and attachment unit interface (AUI) coding and transceiving
circuit with full duplex, testing, isolation, and automatic output selection
IN Paul, Prasun K., Santa Clara, CA, United States
PA National Semiconductor Corporation, Santa Clara, CA, United States (U.S.
corporation)
PI US 5513370 19960430
AI US 1993-113382 19930827 (8)
RLI Continuation-in-part of Ser. No. US 1992-995598, filed on 22 Dec 1992, now patented, Pat. No. US 5446914
DT Utility
LN.CNT 1554
INCL INCLM: 395/800.000
INCLS: 364/222.200; 364/229.500; 364/232.930; 364/239.400; 364/240.100; 364/239.900; 364/242.000; 364/242.500; 364/251.400; 364/264.700; 364/266.400; 364/269.000; 364/271.400; 364/DIG.001; 364/DIG.002; 370/085.300; 340/825.010
NCL NCLM: 709/249.000
NCLS: 340/825.010; 370/239.000
IC [6]
ICM: G06F011-14
ICS: G06F013-40; G06F013-376; G06F015-20
EXF 395/800; 395/750; 395/200; 395/84; 395/325; 395/90; 395/575; 395/200.06;
395/200.1; 395/200.02; 395/650; 370/13; 370/14; 370/16; 370/85.3; 370/79; 370/85.5; 370/85.2; 370/85.7; 370/94.1; 370/85.11; 371/11.2; 371/16.3; 340/825.34; 340/825.13; 340/825.01; 340/825.5; 340/825.36; 340/825.22; 364/DIG.1; 364/DIG.2; 364/580; 375/224; 375/317

L6 ANSWER 4 OF 14 USPATFULL
AN 96:8203 USPATFULL
TI Computer-aided design system
IN Fujita, Shigehisa, Saitama, Japan
Sato, Kenji, Saitama, Japan
PA Honda Giken Kogyo Kabushiki Kaisha, Tokyo, Japan (non-U.S. corporation)
PI US 5487021 19960123
AI US 1993-84073 19930630 (8)
PRAI JP 1992-173522 19920630
DT Utility
LN.CNT 587
INCL INCLM: 364/578.000
NCL NCLM: 395/500.010
IC [6]
ICM: G06F017-00
EXF 364/578; 364/571; 364/490; 364/488; 364/489; 364/476; 364/474.24; 364/560; 395/123; 395/141; 395/134

L6 ANSWER 5 OF 14 USPATFULL
AN 95:79048 USPATFULL
TI Twisted pair and attachment unit interface (AUI) coding and transceiving

circuit with full duplex, testing, and isolation modes
IN Paul, Prasun K., Santa Clara, CA, United States
De Souza, Edwin, Cupertino, CA, United States
PA National Semiconductor Corporation, DE, United States (U.S. corporation)
PI US 5446914 19950829
AI US 1992-995598 19921222 (7)
DT Utility
LN.CNT 1325
INCL INCLM: 395/800.000
INCLS: 364/221.400; 364/221.700; 364/221.800; 364/229.500; 364/230.500; 364/231.400; 364/231.700; 364/237.800; 364/237.900; 364/238.900; 364/239.700; 364/242.000; 364/211.100; 364/265.100; 364/265.200; 364/264.500
NCL NCLM: 709/235.000
IC [6]
ICM: G06F007-02
ICS: G06F009-02; G06F011-16; G06F015-20
EXF 395/20; 395/325; 395/250; 395/275; 395/800; 395/750; 395/575; 395/550; 364/DIG.1; 364/DIG.2; 370/13.1; 370/85.13; 370/85.6; 370/85.9; 370/85.11; 370/85.12; 370/24; 370/26; 370/85.3; 370/91; 370/15; 370/123; 370/75; 370/85.1; 370/85.2; 371/57.2; 371/62; 371/68.2; 371/57.1; 340/825.05; 340/825.12

L6 ANSWER 6 OF 14 USPATFULL
AN 94:49628 USPATFULL
TI Method and apparatus for video signal encoding, decoding and monitoring
IN Copriviza, Robert C., Tarzana, CA, United States
Dubin, Arnold M., Calabasas, CA, United States
Ackerman, Edward B., Encino, CA, United States
Wood, Jackson B., Tarzana, CA, United States
Eakins, Jeffrey S., Claremont, CA, United States
Harmon, David D., Torrance, CA, United States
PA Airtrax, Calabasas, CA, United States (U.S. corporation)
PI US 5319453 19940607
AI US 1989-370399 19890622 (7)
DT Utility
LN.CNT 3096
INCL INCLM: 348/006.000
INCLS: 348/473.000; 348/476.000
NCL NCLM: 346/006.000
NCLS: 348/001.000; 348/473.000; 348/476.000; 455/002.000
IC [5]
ICM: H04N007-087
EXF 358/147; 358/146; 358/142; 358/141; 358/160; 358/86; 358/335; 358/188; 358/143; 358/144; 358/145; 358/83; 360/14.1; 360/14.2; 360/14.3; 360/33.1; H04N007-08; H04N007-087; H04N007-16; H04N007-167; H04N007-175; H04N007-18

L6 ANSWER 7 OF 14 USPATFULL
AN 91:105215 USPATFULL
TI Web registration control system
IN Sainio, Jeffrey W., Hartland, WI, United States
PA Quad/Tech, Inc., Pewaukee, WI, United States (U.S. corporation)
PI US 5076163 19911231
AI US 1989-391784 19890809 (7)
RLI Continuation of Ser. No. US 1986-849095, filed on 7 Apr 1986, now patented, Pat. No. US 4887530
DT Utility
LN.CNT 1927
INCL INCLM: 101/181.000
INCLS: 101/486.000
NCL NCLM: 101/181.000
NCLS: 101/486.000; 347/116.000

IC [5]
ICM: B41F005-06
EXF 101/181; 101/486; 101/DIG.36; 101/211; 101/485; 226/2; 226/3; 226/27; 226/28; 226/29; 226/24; 226/15-18; 226/30-33; 226/45; 250/548; 250/559; 250/561; 250/557; 250/571; 250/226

L6 ANSWER 8 OF 14 USPATFULL
AN 91:78768 USPATFULL
TI Dynamic leak detector
IN Cohrs, Gary D., Tempe, AZ, United States
PA Calibron Systems, Inc., Scottsdale, AZ, United States (U.S. corporation)
PI US 5052212 19911001
AI US 1989-423842 19891018 (7)
RLI Continuation-in-part of Ser. No. US 1988-259847, filed on 19 Oct 1988, now abandoned
DT Utility
LN.CNT 749
INCL INCLM: 073/003.000
INCLS: 073/047.000; 073/198.000
NCL NCLM: 073/001.170
NCLS: 073/047.000; 073/198.000
IC [5]
ICM: G01F025-00
EXF 073/3; 073/47; 073/198

L6 ANSWER 9 OF 14 USPATFULL
AN 89:99804 USPATFULL
TI Web registration control system
IN Sainio, Jeffrey W., Hartland, WI, United States
PA Quad/Tech, Inc., Pewaukee, WI, United States (U.S. corporation)
PI US 4887530 19891219
AI US 1986-849095 19860407 (6)
DT Utility
LN.CNT 1930
INCL INCLM: 101/181.000
INCLS: 101/486.000
NCL NCLM: 101/181.000
NCLS: 101/486.000
IC [4]
ICM: B41F005-06
EXF 101/181; 101/248; 101/DIG.12; 101/211; 101/426; 101/485; 101/486; 226/2;
226/3; 226/27; 226/28; 226/29; 226/24; 226/30-33; 226/15-18; 226/45; 250/548; 250/559; 250/561; 250/557; 250/571; 250/226

L6 ANSWER 10 OF 14 USPATFULL
AN 89:9729 USPATFULL
TI Universal peripheral controller self-configuring bootloadable ramware
IN Klashka, John A., North Andover, MA, United States
Kaufman, Sidney L., Stoughton, MA, United States
Kowal, Krzysztof A., Framingham, MA, United States
Lewis, Richard P., Sandown, NH, United States
Raisbeck, Susan L., Chelmsford, MA, United States
McNamara, Jr., John L., Tewksbury, MA, United States
PA Honeywell Bull Inc., Minneapolis, MN, United States (U.S. corporation)
PI US 4803623 19890207
AI US 1986-925431 19861031 (6)
DT Utility
LN.CNT 1150
INCL INCLM: 364/200.000
NCL NCLM: 710/008.000
IC [4]
ICM: G06F013-10
EXF 364/200MSFile; 364/900MSFile

L6 ANSWER 11 OF 14 USPATFULL
AN 89:1461 USPATFULL
TI Method for monitoring the performance of the head-disk-interface and device for preventing data losses due to magnetic head-disk-interferences
IN Ertingshausen, Friedrich, Nieder-Olm, Germany, Federal Republic of
Mehrens, Jens D., Ober-Olm, Germany, Federal Republic of
PA International Business Machines Corporation, Armonk, NY, United States (U.S. corporation)
PI US 4795981 19890103
AI US 1987-33384 19870331 (7)
PRAI DE 1986-104599 19860404
DT Utility
LN.CNT 462
INCL INCLM: 324/454.000
INCLS: 360/103.000
NCL NCLM: 324/454.000
NCLS: 360/103.000
IC [4]
ICM: G01N027-60
EXF 324/452; 324/454; 324/457; 324/113; 324/210; 324/212; 360/31; 360/103; 360/102; 360/75; 360/105

L6 ANSWER 12 OF 14 USPATFULL
AN 87:26886 USPATFULL
TI CMOS memory margining control circuit for a nonvolatile memory
IN Eby, Michael D., Centerville, OH, United States
PA NCR Corporation, Dayton, OH, United States (U.S. corporation)
PI US 4658380 19870414
AI US 1986-834995 19860228 (6)
DT Utility
LN.CNT 486
INCL INCLM: 365/201.000
NCL NCLM: 365/201.000
IC [4]
ICM: G11C029-00
ICS: G11C011-40
EXF 371/21; 365/201

L6 ANSWER 13 OF 14 USPATFULL
AN 81:34886 USPATFULL
TI I/O Interrupt sequencing for real time and burst mode devices
IN Adams, Jr., Robert L., Kingston, NY, United States
Grant, Carl H., Woodstock, NY, United States
Stevens, Karl W., Saugerties, NY, United States
PA International Business Machines Corporation, Armonk, NY, United States (U.S. corporation)
PI US 4275440 19810623
AI US 1978-948070 19781002 (5)
DT Utility
LN.CNT 358
INCL INCLM: 364/200.000
NCL NCLM: 710/048.000
IC [3]
ICM: G06F003-00
EXF 364/200MSFile; 364/900MSFile

L6 ANSWER 14 OF 14 USPATFULL
AN 74:47884 USPATFULL
TI TRANSPONDER FOR METER READING TELEMETERING SYSTEM
IN Barsellotti, John Anthony, Guelph, Ontario, Canada
Laliccia, Federico Riccardo, Guelph, Ontario, Canada
Litster, John Fraser, Guelph, Ontario, Canada
PA International Standard Electric Corporation, New York, NY, United States
(U.S. corporation)

PI US 3842206 19741015
AI US 1972-308973 19721124 (5)
PRAI CA 1971-130329 19711216
DT Utility
LN.CNT 634
INCL INCLM: 179/002.000A
NCL NCLM: 379/106.070
IC [1]
ICM: H04M011-00
EXF 179/2A; 340/150; 340/151; 340/172.5; 307/85; 307/86

1. 5,748,198, May 5, 1998, Polygon data conversion device, three-dimensional simulator apparatus, and polygon data conversion method; Masaki Takeda, et al., 345/441 [IMAGE AVAILABLE]

2. 5,737,518, Apr. 7, 1998, Method and apparatus for testing an object management system; Douglas M. Grover, et al., 714/38 [IMAGE AVAILABLE]

3. 5,598,562, Jan. 28, 1997, System and method for adding new waitable object types to object oriented computer operating system; David N. Cutler, et al., 709/104, 107 [IMAGE AVAILABLE]

4. 5,486,998, Jan. 23, 1996, Process stabilizing process controller; Ronald Corso, 364/152, 157; 706/23 [IMAGE AVAILABLE]

5. 5,327,579, Jul. 5, 1994, Scanning systems using tree structures; Tosio Kondo, 712/11; 364/229.41, 251.6, DIG.1 [IMAGE AVAILABLE]

6. 5,201,046, Apr. 6, 1993, Relational database management system and method for storing, retrieving and modifying directed graph data structures; Robert N. Goldberg, et al., 707/100; 364/236.2, 239, 242.94, 243, 243.4, 243.41, 252, 254, 254.6, 280, 280.2, 281.3, 281.7, 282.1, 283.4, DIG.1; 707/101 [IMAGE AVAILABLE]

7. 4,975,829, Dec. 4, 1990, Communication interface protocol; Thomas J. Clarey, et al., 395/500.45; 364/231, 235, 236.2, 236.3, 240, 240.8, 274, 280, 280.9, 282.2, 284, 284.2, 284.3, DIG.1; 395/500.41, 500.42; 709/301 [IMAGE AVAILABLE]

=> d l10 1-3 kwic

US PAT NO: 5,748,198 [IMAGE AVAILABLE] L10: 1 of 7

SUMMARY:

BSUM(4)

Various **types** of 3D simulator apparatus that are used in applications such as 3D games or piloting simulators for aircraft or other moving **objects** are known in the art. With such a 3D simulator **apparatus**, image **information** relating to a 3D object 300 shown in FIG. 23A is previously **stored** within the apparatus. In this case, the 3D object 300 depicts an element such as scenery that can be seen. . . The player 302 specifies operations such as rotation or forward motion through a control panel 304, and the present apparatus **performs predetermined** 3D computation **processing** on the basis of the resultant operating signals. More specifically, computations are first performed to determine what changes take place. . .

US PAT NO: 5,737,518 [IMAGE AVAILABLE] L10: 2 of 7

DETDESC:

DETD(34)

Certain **types** of **objects** in object management system 60 can be implemented as tables. A table can include zero or more rows, each row comprising various **objects**. Methods for accessing the **objects** in

a row of a table in network protocols such as SNMP are known to those skilled in the art and will not be described herein. FIG. 7 illustrates a flow diagram for testing table **objects** under the SNMP protocol according to one preferred embodiment of this invention. References to the MIB list in FIG. 7 refer to the list of table **objects** obtained from MIB output file 44 by MIB parser 46. After a table object is selected by a user, test. . . management system 60 to obtain an entry from the table at block 139. At blocks 140 and 141, test signal **generator** 48 waits a **predetermined** amount of time for a response. After waiting the predetermined amount of time without **receiving** a response, test signal generator 48 returns an error at block 142. If a response is **received** within the predetermined amount of time, the object identifier of the returned object can be compared to that of the. . . which comprises a series of digits and periods which identify the object's location in the MIB structure. Because a MIB **structure** is a **tree structure**, **objects** within a table will have a nearly identical object identifier as the table object itself. If the object does not. . . of the table, as indicated in MIB output file 44 were tested. If test signal generator 48 determines that additional **objects** remain in the table, but were not returned from object management system 60, an error is returned at block 149.. . .

US PAT NO: 5,598,562 [IMAGE AVAILABLE] L10: 3 of 7

CLAIMS:

CLMS(1)

What is claimed is:

1. For use in a computer system having a central processing unit and memory means for **storing** data and data **structures**, an operating system comprising:

a multiplicity of different **types** of **objects** comprising data **structures stored** in said memory means;

said multiplicity of data **structures** including a multiplicity of different **types** of synchronization data **structures**;

said multiplicity of different **types** of **objects** including waitable object **types**, each of a multiplicity of said **objects** comprising waitable **objects** incorporating one of said synchronization data **structures**; each synchronization **data structure** incorporated in one of said waitable **objects** enabling an operating system handled thread of execution to wait on said waitable object, said synchronization **data structure** including status means denoting the status of said synchronization **data structure** as Signalled or Unsignalled;

waitable object generating means, responsive to requests from operating system handled threads of execution, for creating ones of said waitable **objects**; wherein, said waitable object generating means **stores** said synchronization **data structure** incorporated in each waitable object at a memory location determinable by a uniform rule applicable to all of said **types** of waitable **objects**;

wait requesting means for suspending a specified operating system handled thread of execution until the status of a specified one of said waitable **objects** is Signalled;

wait service means for changing the status of a specified one of said waitable **objects** to Signalled;

a multiplicity of object type descriptors, each object type descriptor means having means for specifying the format of a distinct one of said different **types** of said **objects**;

a multiplicity of object service means, including an object service means for each said object type for **performing predefined operations** on **objects** of said object type; each said object type descriptor means including means for specifying one of said object service means corresponding to one of said object **types**; and

means for adding to said operating system an additional object type

descriptor means corresponding to an additional type of waitable. . .
specifying one of said object service means for said additional waitable object type and for specifying one of said different **types** of synchronization **data structure** to be incorporated in waitable **objects** of said additional waitable object type;
whereby an additional waitable object type can be added to said operating system by adding. . .

CLAIMS:

CLMS(3)

3. For use in a computer system having a central processing unit and memory means for **storing** data and data **structures**, a method of operating the computer system comprising the steps of:
providing a multiplicity of different **types** of synchronization data **structures**;
providing a multiplicity of **objects** comprising data **structures**, each object being of one of a multiplicity of different object **types**; said multiplicity of different **types** of **objects** including waitable object **types**, each waitable object incorporating one of said multiplicity of different **types** of synchronization data **structures**, enabling operating system handled threads of execution to wait on each said waitable object;
denoting the status of each of said waitable object as Signalled or Unsignalled;
creating ones of said waitable **objects** in response to requests from operating system handled threads of execution, including **storing** said synchronization **data structure** incorporated in each waitable object at a memory location determinable by a uniform rule applicable to all of said waitable object **types**;
suspending a specified operating system handled thread of execution until the status of a specified one of said waitable **objects** is Signalled;
changing the status of a specified one of said waitable **objects** to Signalled;
specifying the format of a distinct one of said different **types** of said **objects** in an object type descriptor specifying an object type service means corresponding to one of said object **types**;
performing predefined operations on **objects** of each said object type using object service means for said each object type; and
generating an additional waitable object type, including
specifying object service means for said additional waitable object type, and
specifying one of said different **types** of synchronization data **structures** to be incorporated in waitable **objects** of said additional waitable object type;
whereby an additional waitable object type can be added to said computer system.

CLAIMS:

CLMS(13)

13. . . . of modifying an operating system for use in a computer system having a central processing unit and memory means for **storing** data and data **structures**, comprising the steps of:
(1) installing an initial operating system that includes:
(1a) a process manager for managing the execution of a plurality of processes;
(1b) a multiplicity of **objects**, each of said **objects** comprising an instance of a **data structure**, wherein said multiplicity of **objects** are partitioned into a multiplicity of object **types**, including waitable object **types**;
(1c) for each object type, an object type descriptor and at least one object **service** means for **performing predefined**

operations on **objects** of the object type; each said object type descriptor specifying the format of **objects** of a corresponding one of said object **types** and including means for specifying one of said object service means associated with said corresponding object type;

(1d) a multiplicity of different **types** of synchronization data **structures**, each synchronization **data structure** having a status of Signalled or Unsignalled;

(1e) a multiplicity of said **objects** comprising waitable **objects**, each having embedded therein one of said synchronization data **structures**, thereby enabling ones of said processes to wait on said waitable object;

(1f) waitable object generating means, responsive to requests from said processes, for creating new instances of waitable **objects**; wherein, said waitable object generating means **stores** said synchronization **data structure** embedded in each waitable object at a memory location determinable by a uniform rule applicable to all of said waitable **objects types**;

(1g) wait requesting means for suspending a specified one of said processes until the status of a specified one of said synchronization data **structures** and its associated waitable object is Signalled;

(1h) wait service means for changing the status of a specified one of said synchronization data **structures** and its associated waitable object to Signalled;

(2) adding to said initial operating system an additional waitable object type, said. . .

adding to said initial operating system object service means for said additional waitable object type;

(2b) specifying one of said different **types** of synchronization data **structures** to be associated with waitable **objects** of said additional waitable object type; and

(2c) generating an additional object type descriptor for said additional object type;

whereby an. . .

CLAIMS:

CLMS(14)

14. An operating system for use in a computer system having a central processing unit and memory means for **storing** data and data **structures**, comprising:

(1) an initial operating system, **stored** in said memory means, that includes:

(1a) a process manager for managing the execution of a plurality of processes;

(1b) a multiplicity of **objects**, each of said **objects** comprising an instance of a **data structure**, wherein said multiplicity of **objects** are partitioned into a multiplicity of object **types**, including waitable object **types**;

(1c) for each object type, an object type descriptor and at least one object **service** means for **performing predefined operations** on **objects** of the object type; each said object type descriptor specifying the format of **objects** of a corresponding one of said object **types** and including means for specifying one of said object service means associated with said corresponding object type;

(1d) a multiplicity of different **types** of synchronization data **structures**, each synchronization **data structure** having a status of Signalled or Unsignalled;

(1e) a multiplicity of said **objects** comprising waitable **objects**, each having embedded therein one of said synchronization data **structures**, thereby enabling ones of said processes to wait on said waitable object;

(1f) waitable object generating means, responsive to requests from said processes, for creating new instances of waitable **objects**;

wherein, said waitable object generating means **stores** said synchronization **data structure** embedded in each waitable object at a memory location determinable by a uniform rule applicable to all of said waitable **objects types**;

(1g) wait requesting means for suspending a specified one of said processes until the status of a specified one of said synchronization data **structures** and its associated waitable object is Signalled;

(1h) wait service means for changing the status of a specified one of said synchronization data **structures** and its associated waitable object to Signalled;

(2) means for adding to said initial operating system an additional waitable object. . . means for adding:

(2a) object service means for said additional waitable object type;

(2b) means for specifying one of said different **types** of synchronization data **structures** to be associated with waitable **objects** of said additional waitable object type; and

(2c) an additional object type descriptor for said additional object type;

whereby an additional. . .

CLAIMS:

CLMS(15)

15. . . . of modifying an operating system for use in a computer system having a central processing unit and memory means for **storing** data and data **structures**, comprising the steps of:

(1) providing an initial operating system that includes:

(1a) thread management means for managing the execution of a plurality of threads of execution;

(1b) a multiplicity of **objects**, each of said **objects** comprising an instance of a **data structure**, wherein said multiplicity of **objects** are partitioned into a multiplicity of object **types**, including waitable object **types**;

(1c) for each object type, an object type descriptor and at least one object **service** means for **performing predefined operations** on **objects** of the object type; each said object type descriptor specifying the format of **objects** of a corresponding one of said object **types** and including means for specifying one of said object service means associated with said corresponding object type;

(1d) a multiplicity of different **types** of synchronization data **structures**, each synchronization **data structure** having a status of Signalled or Unsignalled;

(1e) a multiplicity of said **objects** comprising waitable **objects**, each having incorporated therein one of said synchronization data **structures**, thereby enabling ones of said threads of execution to wait on said waitable object;

(1f) waitable object generating means, responsive to requests from said threads of execution, for creating new instances of waitable **objects**; wherein, said waitable object generating means **stores** said synchronization **data structure** incorporated in each waitable object at a memory location determinable by a uniform rule applicable to all of said **types** of waitable **objects**;

(1g) wait requesting means for suspending a specified one of said threads of execution until the status of a specified one of said synchronization data **structures** and its associated waitable object is Signalled;

(1h) wait service means for changing the status of a specified one of said synchronization data **structures** and its associated waitable object to Signalled;

(2) adding to said initial operating system an additional waitable object type, said. . .

adding to said initial operating system object service means for said additional waitable object type;

(2b) specifying one of said different **types** of synchronization data

structures to be associated with waitable **objects** of said additional waitable object type; and
(2c) generating an additional object type descriptor for said additional object type;
whereby an. . .

CLAIMS:

CLMS(16)

16. An operating system for use in a computer system having a central processing unit and memory means for **storing** data and data **structures**, comprising:
(1) an initial operating system, **stored** in said memory means, that includes:
(1a) thread management means for managing the execution of a plurality of threads of execution;
(1b) a multiplicity of **objects**, each of said **objects** comprising an instance of a **data structure**, wherein said multiplicity of **objects** are partitioned into a multiplicity of object **types**, including waitable object **types**;
(1c) for each object type, an object type descriptor and at least one object **service** means for **performing predefined operations** on **objects** of the object type; each said object type descriptor specifying the format of **objects** of a corresponding one of said object **types** and including means for specifying one of said object service means associated with said corresponding object type;
(1d) a multiplicity of different **types** of synchronization data **structures**, each synchronization **data structure** having a status of Signalled or Unsignalled;
(1e) a multiplicity of said **objects** comprising waitable **objects**, each having incorporated therein one of said synchronization data **structures**, thereby enabling ones of said threads of execution to wait on said waitable object;
(1f) waitable object generating means, responsive to requests from said threads of execution, for creating new instances of waitable **objects**; wherein, said waitable object generating means **stores** said synchronization **data structure** incorporated in each waitable object at a memory location determinable by a uniform rule applicable to all of said **types** of waitable **objects**;
(1g) wait requesting means for suspending a specified one of said threads of execution until the status of a specified one of said synchronization data **structures** and its associated waitable object is Signalled;
(1h) wait service means for changing the status of a specified one of said synchronization data **structures** and its associated waitable object to Signalled;
(2) means adding to said initial operating system an additional waitable object type,. . . including means for adding:
(2a) service means for said additional waitable object type;
(2b) means for specifying one of said different **types** of synchronization data **structures** to be associated with waitable **objects** of said additional waitable object type; and
(2c) an additional object type descriptor for said additional object type;
whereby an additional. . .

CLAIMS:

CLMS(22)

22. For use in a computer system, a method of managing resources of the computer system, the resources partitioned into resource **classes**, the resource **classes** being divided into waitable and non-waitable **classes**, the method comprising the steps of:

for each resource, allocating **storage** for a resource **data structure** for use in managing said each resource, said resource data **structures** being partitioned into object **classes** corresponding to said resource **classes**, said object **classes** including waitable and non-waitable object **classes**, each object class comprising a corresponding **data structure** format and at least one corresponding object **service** routine for **performing predefined operations** on resource data **structures** of said object class, the **data structure** formats for said object **classes** sharing a common header format;

for each said object class, allocating **storage** for an object type descriptor corresponding to said object class, each object type descriptor comprising **data** indicating said **data structure** format and said at least one object service routine corresponding to said object class;

setting a type pointer member in the header of each said resource **data structure** to point to the corresponding object type descriptor;

launching a plurality of processes for execution;

incorporating in each said resource **data structure** corresponding to a waitable one of said resources a synchronization **data structure**, each synchronization **data structure** having a status of Signalled or Unsignalled and having a datum for indicating processes waiting on said synchronization **data structure**; said incorporating step including **storing** said synchronization **data structure** incorporated in each resource **data structure** at a memory location determinable by a uniform rule applicable to all of said waitable object **classes**;

when one of said processes requests service from one of said waitable resources, setting to Unsignalled the status of the synchronization **data structure** associated with the resource **data structure** corresponding to said one waitable resource, and suspending said one process until the status of the associated synchronization **data structure** is Signalled; and

when a wait condition for said waitable resource is satisfied, changing the status of said corresponding synchronization **data structure** to Signalled.

CLAIMS:

CLMS(24)

24. . . . during which the process must wait pending completion of the service, the steps of the method comprising:

locating a resource **data structure**, corresponding to the one resource from the service requested, from among a plurality of resource data **structures** each corresponding to one of said plurality of resources and used by said operating system for use in managing the corresponding resource, each of said resource data **structures** belonging to one of a plurality of object **classes**, each object class having a **data structure** format and at least one object **service** routine for **performing predefined operations** on resource data **structures** of said each object class;

locating a synchronization **data structure** for said resource **data structure** by means of a rule uniformly applicable to said resource data **structures** of all said object **classes**, said synchronization **data structure** specifying a synchronization protocol associated with said service;

suspending execution of the process according to said synchronization protocol, and **storing** in association with said synchronization **data structure** process control **data** for resuming execution of the process; and

when said synchronization protocol is satisfied, resuming execution of the process according to said. . .